EECE 4712 - Embedded Systems
Arduino Shield Project
Spring 2025

Regular Section:

Design a shield for an Arduino uno that acts as an I2C bus interface for sensors to plug into. I2C sensors are quite common, especially in the breakout board Arduino space. Interfacing to the breakout boards can leave you with a wiring nightmare if you're not careful. Use KiCad (Version 6.0 or greater) to create a schematic and layout for an Arduino shield with the following requirements.

Minimum Hardware Requirements:

- I2C bus must connect to SCL and SDA pins of the Arduino uno
- I2C bus headers be 0.1" pitch
- I2C bus should have 4 pins per position (PWR, GND, SCL, SDA)
- I2C bus should have 8 positions for sensors to plug into
- Must fit perfectly onto the Arduino uno, socketing into all of it's headers
- Power the I2C devices independently of the Arduino with a 5V linear regulator
- Keep the 5V regulator power separate from the 5V Arduino power
- Power the 5V regulator from the VIN pin from the Arduino uno
- Use stackable headers like: https://www.digikey.com/en/products/detail/sparkfun-electronics/PRT-11417/6161755
- Ensure the silkscreen layer is organized

Some notes about power pins: Using pin headers instead of sockets seems cool at first, but when you accidentally bridge power and ground together things can get heated. JST-XH connectors are recessed and can stop most accidental bridging from happening and are somewhat more professional than the regular 0.1" pin headers in some cases. But this can be more inconvenient than regular 0.1" headers and they take up much more room

on the board. You could use the regular 0.1” female headers, but the downsides to this are plugging into female headers makes the connector twice as long and a bit less secure. The choice out of these three is up to you.

Honors/Grad Section:

In addition to the regular section requirements. This section will need to add an I2C “selection.” Sometimes we will need to use many of the same sensors, and they may all default to the same I2C address. To combat this, some sensors will have a shutdown pin on them to turn the sensor off so you can reprogram the address of each sensor one at a time. Some sensors do not have this luxury.

Minimum Hardware Requirements:

- Everything included in the regular section plus:
- Use a serial to parallel shift register to control the IO expansion capabilities of addressing each I2C position’s power control circuit (take note of output type. Make sure it can drive the MOSFET)
- Shift register must have a register and serial clock on it. (The SRCLK will shift in data internally without changing the output, clocking in through RCLK will output what’s inside the internal register)

Example of a sensor with a shutdown pin:
https://learn.adafruit.com/adafruit-vl53l0x-micro-lidar-distance-sensor-breakout

Challenge Mode:

Add the ability to level shift between 5V and 3.3V on the SDA and SCL lines on the I2C bus so you can control devices running on a different voltage standard. Some I2C sensors will have the capability to accept multiple voltage levels from 3-5V. Some do not.

Minimum Hardware Requirements:

- Everything included in the regular and honors section plus:
- Make 4 of the I2C positions 3.3V
- Make 4 of the I2C positions 5V
- A 3.3V regulator will also need to be added for this
- Keep the 3.3V regulator power separate from the 3.3V Arduino power
- Still use the SDA and SCL from the Arduino uno as the only controlling I2C interface from the Arduino

I would recommend taking a stab at this problem and see what solutions you can come up with.

Freeform Mode:

Don’t let the minimum requirements stop you from your creativity. If you want to add something cool to this project, do it.